

REMARKS

This preliminary amendment is submitted to clarify terminology corresponding to a referenced item provided in the specification and drawings. The clarification refers to commonly utilized standard terminology. No new matter is added.

Consideration and entry of this Amendment is respectfully requested. If there are any additional charges with respect to this Preliminary Amendment or otherwise, please charge them to Deposit Account No. 06-1130 maintained by applicant's attorney.

Respectfully submitted,  
STEVEN J. KINDER ET AL

CANTOR COLBURN LLP  
Applicant's Attorneys

By Marisa J. Dubuc  
Marisa J. Dubuc  
Registration No. 46,673  
Customer No. 23413  
Confirmation No. 5881

Date: July 16, 2001  
Address: 55 Griffin Road South, Bloomfield, CT 06002  
Telephone: (860) 286-2929

**Version with Markings to show changes made:**

**IN THE DRAWINGS**

The attached drawing amendments “marked up” in red are submitted for the Examiner’s approval.

In FIGs 1 through 4: Replace “NT” with “WS”

**IN THE DESCRIPTION OF THE PREFERRED EMBODIMENT**

A “marked up” version of the Detailed Description follows:

From page 4, paragraph 3:

FIG. 2 is a diagram depicting a directed acyclic distributed transaction tree. A distributed ~~NT~~ WS node 52 usually has a well-known name that it uses as its branch qualifier (BQUAL) 56, 60. The distributed ~~NT~~ WS node 52 sends syncpoint cues in the form of outbound flow 62 to a subordinate node 54. The outbound flow is received as inbound flow 62 by the subordinate node 54. The TM of the subordinate node 54 examines its registry for the inbound flow’s GTRID before proceeding. If the TM of the subordinate node 54 has not seen the incoming flow’s 62 GTRID it will create a new branch in the registry for this transaction, and add the BQUAL to the inbound branch registry. If the TM of subordinate node 54 has seen the incoming flow 62, it uses the transaction that was created previously

to coordinate updates to protected resources. Thus, the TM considers this reentry as a direct synchronous inbound flow because the distributed NTWS node 52 may flow to subordinate node 54 and to subordinate node 58 repeatedly with no update to BQUAL 60 prior to committing.

From page 5, paragraph 1:

FIG. 3 is a diagram depicting a distributed transaction tree containing a cycle. Node 104 receives two inbound flows 116, 118; one from the NT WS node 102, and one from node 114. When node 104 receives the flow there is the possibility that it will erroneously assume that this is the same branch in the distributed tree that was noted before. If this branch is assumed to be the same as the one previously sent there will be a likelihood that there will be an unwanted sharing of protected resource locks. Further, node 104 may become confused with its syncpoint responsibilities. To correct this confusion, node 104 will receive a prepare flow from the NT WS node 102 and node 114 during the commit processing phase. Consequently, node 104 will be directed to prepare, for which it will drive pre-prepare instructions for all local resource managers, and after preparing will flow a prepare signal to node 110. In sending the prepare signal node 104 also sends its non-incremented BQUAL 106. Similarly, node 110 issues pre-prepare instructions and prepares local resources. Next, node 110 directs the flow to node 114 along with its non-incremented BQUAL 108. During the preparing of resources for node 114, it is possible that updates may flow to node 104. Consequently, the pre-prepare instruction of node 114 causes updates on node 104 and its non-incremented BQUAL 106, which has been previously prepared. Unfortunately, allowing updates on node 104 without incrementing the BQUAL 112 will invariably result in an error.

From page 6, paragraph 1:

FIG. 4 is a diagram depicting a solution to the problem introduced by cyclic distribution transaction trees by introducing a path-sensitive branch registry. Here, node 154 receives an inbound flow 166 from node NT WS 152. Node 154 does not find the inbound flow 166 from node NT WS 152 in its inbound registry. Next, node 154 will associate an indexed, transaction-unique BQUAL 156 (A1) with the inbound flow, where the index (1) indicates the number of times that the transaction has looped through the node 154. Subsequently, node 154 will send a flow 168 and its BQUAL 156 (A1) to node 160. Next, node 160 will receive the inbound flow 168, and associate its own indexed BQUAL 158 (B1) with the inbound indexed BQUAL 156 (A1). Likewise, node 160 sends a flow 170 with its indexed BQUAL 158 (B1) to node 164. The cycle completes when node 164 sends a flow 172 with its indexed BQUAL 162 (C1) to node 154, where node 154 will consult its inbound registry to see that it has not received an inbound flow from node 164 for this transaction, and will create a new BQUAL (A2) with an incremented index (2) that is different for any other index in the registry for that node for that transaction. Therefore, the cyclic flow, NT WS → A → B → C → A has become the acyclic flow NT WS → A1 → B1 → C1 → A2.

From page 6 paragraph 2:

FIG. 5 is a diagram depicting a creation of a path-sensitive registry. Node 202 is a subordinate of node 214 and other nodes based on the inbound flows 218. Node 202 is a superior to node 210. As in FIG. 3, NT WS will deliver a prepare instruction to node 202, in turn node 202 will issue pre-prepare instructions to the local resources. Node 202 will then prepare local resources, and flow prepare to node 210. Next, node 210 flows prepare to node 214. Consequently, after preparing local resources node 214 flows prepare to node 202. Following the prepare of the local resources node 202 then pre-prepares local resources associated with this subordinate transaction. Therefore, the path-sensitive registry prevents the unwanted sharing of database and/or protected resource locks and correctly delivers pre-prepare to objects and prepare to resources. The unwanted sharing of database and/or protected resource locks and correct delivery of pre-prepare objects is achieved by incrementing BQUALs 204, 206, 208 and 212 before they are sent to another node.

From page 7 paragraph 1:

FIG. 6 is a diagram depicting the relationship between transaction managers and their subordinates. As in FIG. 3, NT WS 252 represents a root transaction. The TM 254 of node 260 receives inbound flow 278 from the transaction root 252 and evaluated against the inbound registry of the node. Specifically, an inbound registry contains the node's BQUAL 258 and its GTRID 256. The TM 254 compares incoming syncpoint cues with those stored in the memory. When the TM 254 does not find a matching syncpoint cue it adds the incoming syncpoint cue to its registry and increments its BQUAL and links it to the syncpoint cue prior to sending it to its subordinate(s). In addition, the registry contains the node's BQUAL 258. For example, node 260 sends outbound flow to node 276 and node 274. The TM of a

subordinate node 254 searches its registry to see if the inbound flow's GTRID matches any previously recorded GTRID. If there is no match the TM 254 directs the recording of the new GTRID. If there is a match the TM 254 renames the transaction and sends the newly named flow 282 to its subordinate(s), and so on. Next, the subordinate nodes 274 receives the outbound flow 282 and reports back to the superior node 260, confirming its subordinate status.

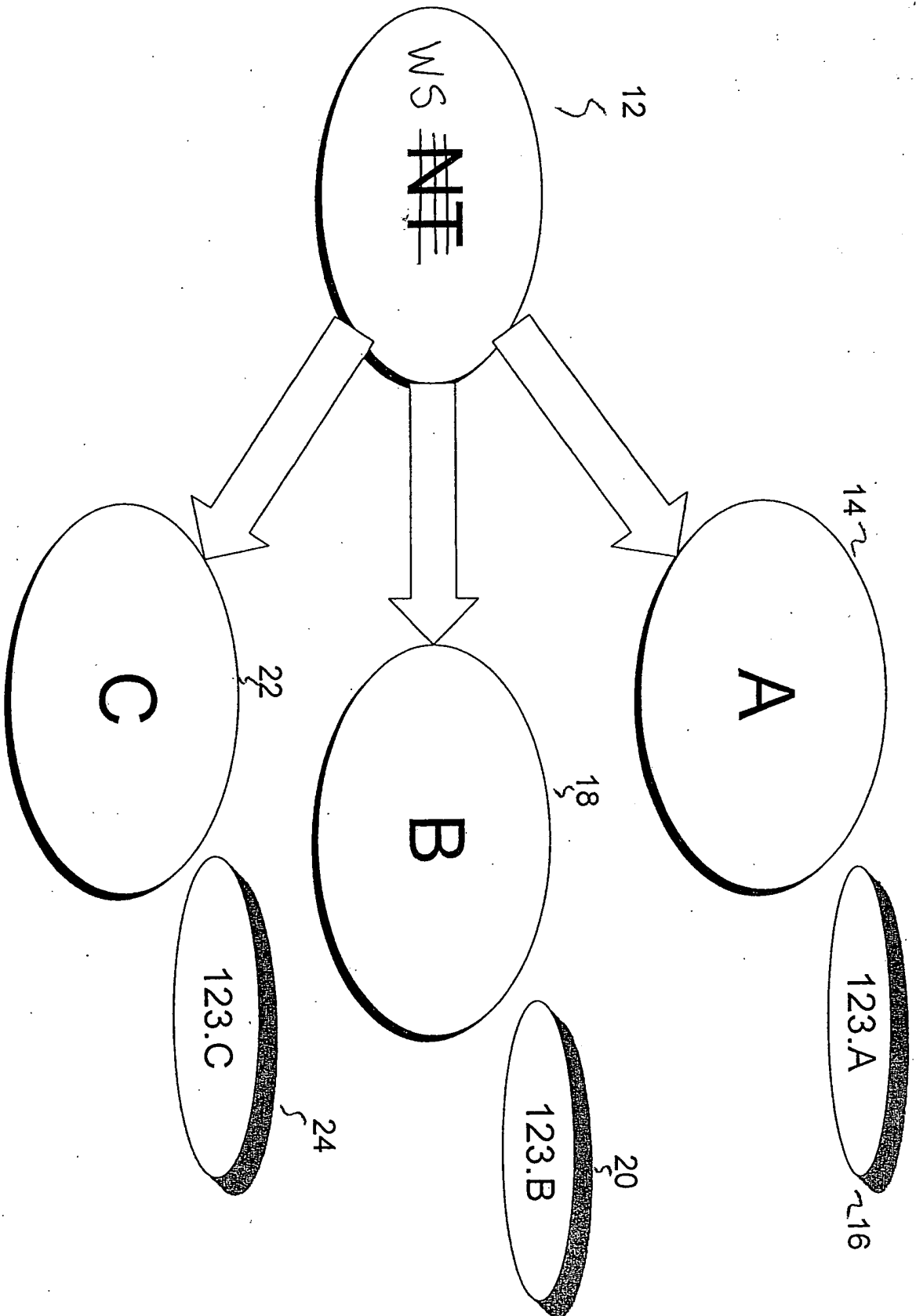


FIG. 1

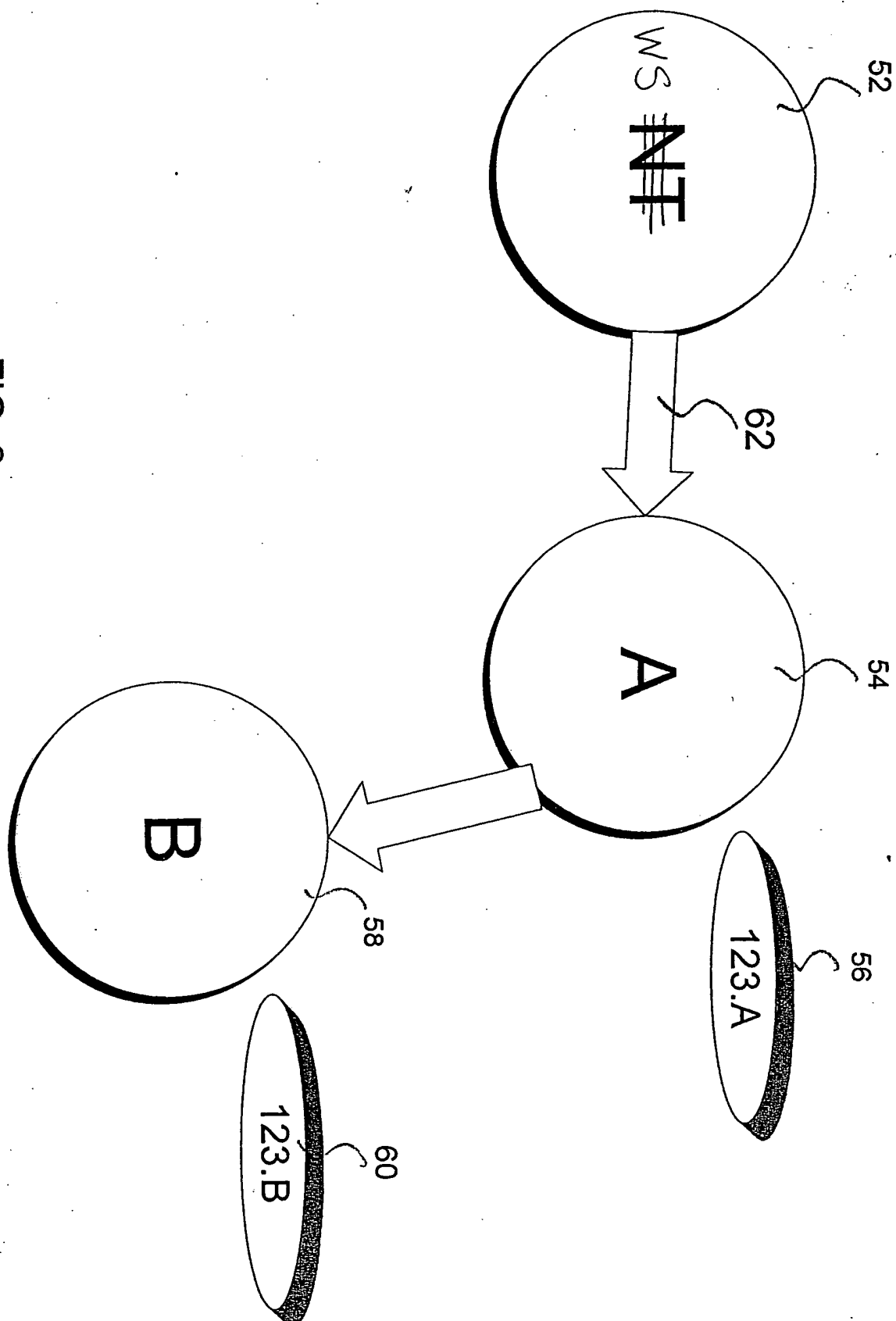


FIG. 2

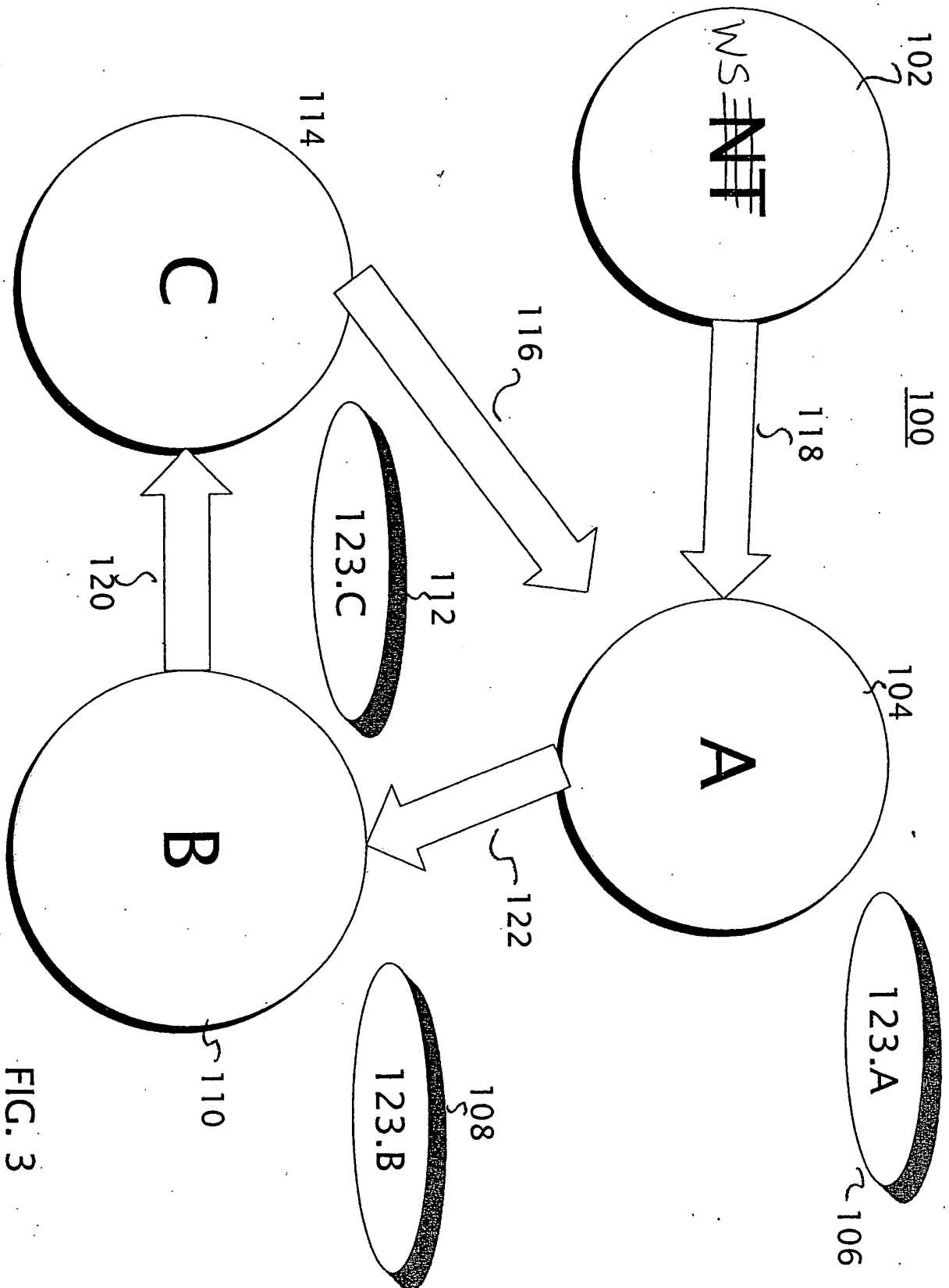


FIG. 3

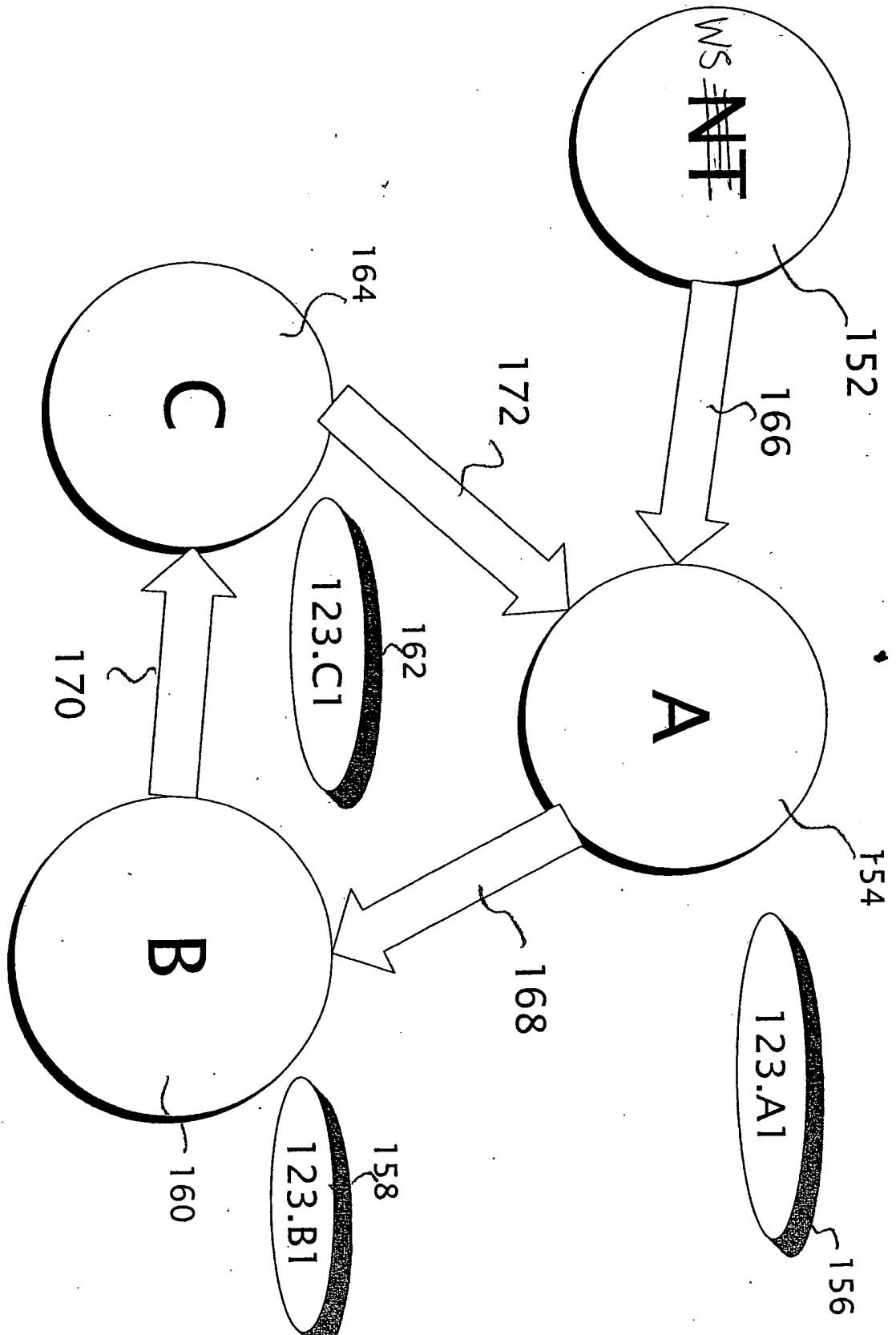


FIG. 4